**Decision tree**

The definition of classifying is categorizing something or someone into a certain group or system based on certain characteristics. An example of classifying is assigning plants or animals into a kingdom and species.  
  
The classification technique is a systematic approach to build classification models from an input dat set. For example, decision tree classifiers, rule-based classifiers, neural networks, support vector machines, and naive Bayes classifiers are different technique to solve a classification problem. Each technique adopts a learning algorithm to identify a model that best fits the relationshio between the attribute set and class label of the input data. Therefore, a key objective of the learning algorithm is to build prdictive model that accurately predict the class labels of previously unkonw records.

**what is Decision tree ?**  
A decision tree is a graphical representation of all the possible solutions to decision based on certain conditions

Decision Tree Classifier is a simple and widely used classification technique. It applies a straitforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time time it receive an answer, a follow-up question is asked until a conclusion about the calss label of the record is reached.  
  
**Determine the best attribute test Condition**  
The decision tree inducing algorithm must provide a method for specifying the test condition for different attribute types as well as an objective measure for evaluating the good ness of each test condition.  
  
First, the specification of an attribute test condition and its corresponding outcomes depends on the attribute types. We can do two-way split or multi-way split, discretize or group attribute values as needed. The binary attributes leads to two-way split test condition. For norminal attributes which have many values, the test condition can be expressed into multiway split on each distinct values, or two-way split by grouping the attribute values into two subsets. Similarly, the ordinal attributes can also produce binary or multiway splits as long as the grouing does not violate the order property of the attribute values. For continuous attributes, the test condition can be expressed as a comparsion test with two outcomes, or a range query. Or we can discretize the continous value into nominal attribute and then perform two-way or multi-way split.  
  
Since there are may choices to specify the test conditions from the given training set, we need use a measurement to determine the best way to split the records. The goal of best test conditions is whether it leads a homogenous class distribution in the nodes, which is the purity of the child nodes before and after spliting. The larger the degree of purity, the better the class distribution.  
  
To determine how well a test condition performs, we need to compare the degree of impurity of the parent before spliting with degree of the impurity of the child nodes after splitting. The larger their differnce, the better the test condition. The measurment of node impurity/purity are:  Gini Index Entropy

example 1

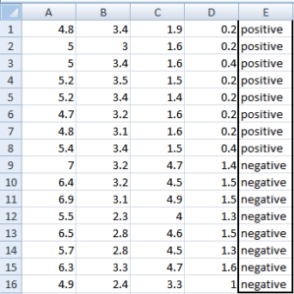
By calculating **entropy measure** of each attribute we can calculate their **information gain**. Information Gain calculates the expected reduction in entropy due to sorting on the attribute. Information gain can be calculated.

To get a clear understanding of calculating **information gain & entropy**, we will try to implement it on a sample data.

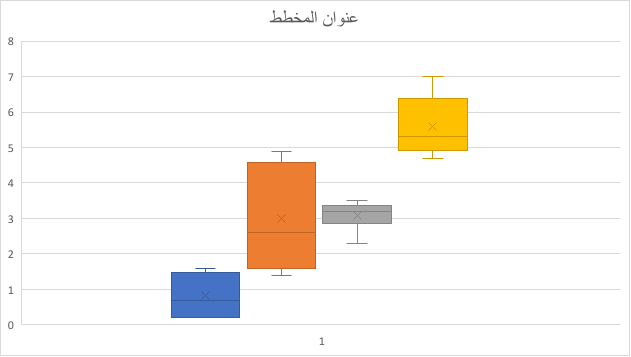
#### Example: Construct a Decision Tree by using “information gain” as a criterion

We are going to use this data sample. Let’s try to use information gain as a criterion. Here, we have 5 columns out of which 4 columns have continuous data and 5th column consists of class labels.

A, B, C, D attributes can be considered as predictors and E column class labels can be considered as a target variable. For constructing a decision tree from this data, we have to convert continuous data into categorical data.

We have chosen some random values to categorize each attribute:

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **C** | **D** |
| >= 5 | >= 3.0 | >= 4.2 | >= 1.4 |
| < 5 | < 3.0 | < 4.2 | < 1.4 |

There are **2 steps for calculating information gain** for each attribute:

1. Calculate entropy of Target.
2. Entropy for every attribute A, B, C, D needs to be calculated. Using information gain formula we will subtract this entropy from the entropy of target. The result is Information Gain.

**The entropy of Target:** We have 8 records with negative class and 8 records with positive class. So, we can directly estimate the entropy of target as 1.

|  |  |
| --- | --- |
| Variable E | |
| Positive | Negative |
| 8 | 8 |

**Calculating entropy using formula:**

E(s) = -1\*( (p(+ve)\*log( p(+ve)) + (p(-ve)\*log( p(-ve)) )  
= -1\*( (8/16)\*log2(8/16)) + (8/16) \* log2(8/16) )  
= 1

#### Information gain for Var A

Var A has value >=5 for 12 records out of 16 and 4 records with value <5 value.

* For Var A >= 5 & class == positive: 5/12
* For Var A >= 5 & class == negative: 7/12
  + Entropy(5,7) = -1 \* ( (5/12)\*log2(5/12) + (7/12)\*log2(7/12)) = 0.9799
* For Var A <5 & class == positive: 3/4
* For Var A <5 & class == negative: 1/4
  + Entropy(3,1) =  -1 \* ( (3/4)\*log2(3/4) + (1/4)\*log2(1/4)) = 0.81128

Entropy(Target, A) = P(>=5) \* E(5,7) + P(<5) \* E(3,1)  
= (12/16) \* 0.9799 + (4/16) \* 0.81128 = 0.937745

#### Gain ( A ) = E( s ) - I( A ) = 1 -   0.937745  = 0.062255

#### Information gain for Var B

Var B has value >=3 for 12 records out of 16 and 4 records with value <5 value.

* For Var B >= 3 & class == positive: 8/12
* For Var B >= 3 & class == negative: 4/12
  + Entropy(8,4) = -1 \* ( (8/12)\*log2(8/12) + (4/12)\*log2(4/12)) = 0.39054
* For VarB <3 & class == positive: 0/4
* For Var B <3 & class == negative: 4/4
  + Entropy(0,4) =  -1 \* ( (0/4)\*log2(0/4) + (4/4)\*log2(4/4)) = 0

Entropy(Target, B) = P(>=3) \* E(8,4) + P(<3) \* E(0,4)  
= (12/16) \* 0.39054 + (4/16) \* 0 = 0.292905

Gain ( B ) = E( s ) - I( B ) = 1 -  0.292905 = 0.707095

#### Information gain for Var C

Var C has value >=4.2 for 6 records out of 16 and 10 records with value <4.2 value.

* For Var C >= 4.2 & class == positive: 0/6
* For Var C >= 4.2 & class == negative:  6/6
  + Entropy(0,6) = 0
* For VarC < 4.2 & class == positive: 8/10
* For Var C < 4.2 & class == negative: 2/10
  + Entropy(8,2) = 0.72193

Entropy(Target, C) = P(>=4.2) \* E(0,6) + P(< 4.2) \* E(8,2)  
= (6/16) \* 0 + (10/16) \* 0.72193 = 0.4512

#### Gain ( C ) = E( s ) - I( C ) = 1 -   0.4512   = 0.5488

#### Information gain for Var D

Var D has value >=1.4 for 5 records out of 16 and 11 records with value <5 value.

* For Var D >= 1.4 & class == positive: 0/5
* For Var D >= 1.4 & class == negative: 5/5
  + Entropy(0,5) = 0
* For Var D < 1.4 & class == positive: 8/11
* For Var D < 14 & class == negative: 3/11
  + Entropy(8,3) =  -1 \* ( (8/11)\*log2(8/11) + (3/11)\*log2(3/11)) = 0.84532

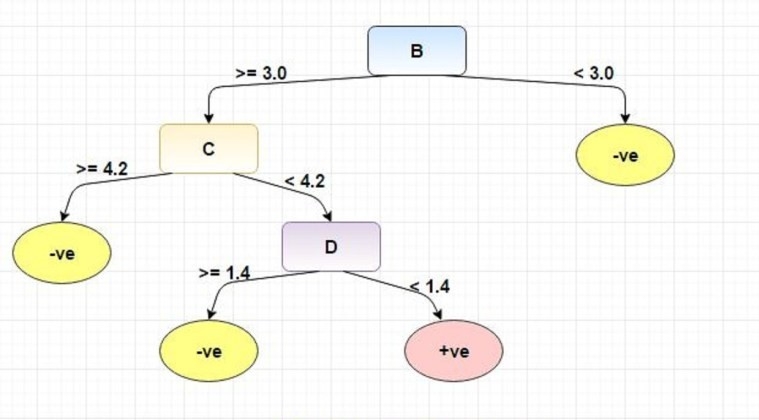
Entropy(Target, D) = P(>=1.4) \* E(0,5) + P(< 1.4) \* E(8,3)  
= 5/16 \* 0 + (11/16) \* 0.84532 = 0.5811575

  Gain ( D ) = E( s ) - I( D ) = 1 -   0.5811575  = 0.411890

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | | **Target** | | | Positive | Negative | | A | >= 5.0 | 5 | 7 | | <5 | 3 | 1 | | Information Gain of A = 0.062255 | | | | | |  |  |  |  | | --- | --- | --- | --- | |  | | **Target** | | | Positive | Negative | | B | >= 3.0 | 8 | 4 | | < 3.0 | 0 | 4 | | Information Gain of B= 0.7070795 | | | | |
| |  |  |  |  | | --- | --- | --- | --- | |  | | **Target** | | | Positive | Negative | | C | >= 4.2 | 0 | 6 | | < 4.2 | 8 | 2 | | Information Gain of C= 0.5488 | | | | | |  |  |  |  | | --- | --- | --- | --- | |  | | **Target** | | | Positive | Negative | | D | >= 1.4 | 0 | 5 | | < 1.4 | 8 | 3 | | Information Gain of D= 0.41189 | | | | |

From the above Information Gain calculations, we can build a decision tree. We should place the attributes on the tree according to their values.

An Attribute with better value than other should position as root and A branch with entropy 0 should be converted to a leaf node. A branch with entropy more than 0 needs further splitting.



  example  2  
  
Outlook    Temperature   Humidity  Windy   Play?  
Sunny         hot         High     Weak    No  
Sunny         hot         High    Strong   No  
Overcast      hot         High     Weak    Yes  
Rainy         mid         High     Weak    Yes  
Rainy         coot        Normal   Weak    Yes  
Rainy         coot        Normal   Strong   No  
Overcast      coot        Normal   Strong  Yes  
Sunny         mid         High     Weak    No  
Sunny         coot        Normal   Weak  Yes  
Rainy         mid         Normal   Weak    Yes  
Sunny         mid         Normal   Strong  Yes  
Overcast      mid         High     Strong  Yes  
Overcast      hot         Normal   Weak    Yes  
Rainy         mid         High    Strong   No  
  
  
E(s) = - 9/14 log2 ( 9/14 ) - 5/14 \* log2( 5/14 )  =0.94  
  
                              yes  
E(Outlook = Sunny    )=-2/5 log2( 2/5 ) -2/5 log2( 2/5 )     =0.971  
E(Outlook = Overcast )=-1 log2( 1 ) -0 log2( 0 )             = 0  
E(Outlook = Rainy    )=-2/5 log2( 2/5 ) -2/5 log2( 2/5 )     =0.971  
I( Outlook )          =5/14 \* 0.971 + 4/14 \* 0 +5/14 \* 0.971 = 0.693  
  
Gain( Outlook  )= E(s) - I( Outlook ) = 0.94 - 0.693 = 0.247  
  
---------------------  
E(Windy   = Weak    )=-6/8 log2( 6/8 ) -2/8 log2( 2/8 )     = 0.811  
E(Windy   = Strong  )=-3/6 log2(3/6 ) -3/6 log2(3/6 )       = 1  
I(Windy   )          =6/14 \*1 + 8/14 \* 0.811                = 0.892  
Gain( Windy  )= E(s) - I( Windy ) = 0.94 - 0.892 = 0.048  
  
---------------------------  
Gain( Temperature   )= 0.94 - 0.911 = 0.029  
  
---------------------------  
Gain( Humidity  )= 0.94 - 0.788 = 0.152  
--------------------------  
Outlook    
{Sunny  { Humidity { normal: yes , high : no  }  } ,   
Overcast  { yes } ,  
 Rainy  { Windy { strong : no , week: yes  } }    }  
  
  
Decision Tree Algorithm Advantages and Disadvantages  
Advantages:  
1.Decision Trees are easy to explain. It results in a set of rules.  
2.It follows the same approach as humans generally follow while making decisions.  
3.Interpretation of a complex Decision Tree model can be simplified by its visualizations.  
Even a naive person can understand logic.  
4.The Number of hyper-parameters to be tuned is almost null.  
Disadvantages:  
1.There is a high probability of overfitting in Decision Tree.  
2.Generally, it gives low prediction accuracy for a dataset as compared to other machine learning algorithms.  
3.Information gain in a decision tree with categorical variables gives a biased response for attributes with greater no. of categories.  
4.Calculations can become complex when there are many class labels.  
  
We are in the midst of an unprecedented surge of interest in machine learning (ML) and artificial  
intelligence (AI) technologies. These tools, which allow computers to make data-derived predictions  
and automate decisions, have become part of daily life for billions of people. Ubiquitous digital  
services such as interactive maps, tailored advertisements, and voice-activated personal assistants  
are likely only the beginning. Some AI advocates even claim that AI’s impact will be as profound  
as “electricity or fire” that it will revolutionize nearly every field of human activity. This enthusiasm  
has reached international development as well    
  
Emerging ML/AI applications promise to reshape healthcare and agriculture  in the developing world. ML and AI show tremendous potential for helping to achieve sustainable development objectives globally. They can  
improve efficiency by automating labor-intensive tasks, or offer new insights by finding patterns in  
large, complex datasets.  
A recent report suggests that AI advances could double economic growth rates and increase labor productivity 40% by 2035

**Decision Tree Code: Implementation with Python**

Car Evaluation Database

The dataset is available at: “<http://archive.ics.uci.edu/ml/datasets/Car+Evaluation>”

0) Import necessary libraries  
Import the necessary modules from specific libraries.  
  
import os  
import numpy as np  
import pandas as pd  
import numpy as np, pandas as pd  
import matplotlib.pyplot as plt  
from sklearn import tree, metrics  
1) Load the data set  
Use pandas module to read the bike data from the file system. Check a few records of the dataset.  
  
data =  
pd.read\_csv('data/car\_quality/car.data',names=['buying','maint','doors','persons','lug\_boot','safety','class'])  
data.head()  
  
  buying maint doors persons lug\_boot safety class  
0 vhigh  vhigh 2     2       small    low    unacc  
1 vhigh  vhigh 2     2       small    med    unacc  
2 vhigh  vhigh 2     2       small    high   unacc  
3 vhigh  vhigh 2     2       med      low    unacc  
4 vhigh  vhigh 2     2       med      med    unacc

2) Check a few information about the data set  
[data.info](http://data.info/)()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1728 entries, 0 to 1727  
Data columns (total 7 columns):  
buying      1728 non-null object  
maint       1728 non-null object  
doors       1728 non-null object  
persons     1728 non-null object  
lug\_boot    1728 non-null object  
safety      1728 non-null object  
class       1728 non-null object  
dtypes: object(7)  
memory usage: 94.6+ KB  
The training dataset has 1728 rows and 7 columns.  
  
There are no missing values in the dataset  
  
3) Identify the target variable

data['class'],class\_names = pd.factorize(data['class'])  
 .  
  
Let’s check the encoded values now.  
  
print(class\_names)  
print(data['class'].unique())  
  
Index([u'unacc', u'acc', u'vgood', u'good'], dtype='object')  
[0 1 2 3]

As we can see the values has been encoded into 4 different numeric labels.  
  
4) Identify the predictor variables and encode any string variables to equivalent integer codes

data['buying'],\_ = pd.factorize(data['buying'])  
data['maint'],\_ = pd.factorize(data['maint'])  
data['doors'],\_ = pd.factorize(data['doors'])  
data['persons'],\_ = pd.factorize(data['persons'])  
data['lug\_boot'],\_ = pd.factorize(data['lug\_boot'])  
data['safety'],\_ = pd.factorize(data['safety'])  
data.head()  
  
  buying maint doors persons lug\_boot safety class  
0       0      0     0     0       0        0      0  
1       0      0     0     0       0        1      0  
2       0      0     0     0       0        2      0  
3       0      0     0     0       1        0      0  
4       0      0     0     0       1        1      0

Check the data types now:  
  
[data.info](http://data.info/)()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1728 entries, 0 to 1727  
Data columns (total 7 columns):  
buying      1728 non-null int64  
maint       1728 non-null int64  
doors       1728 non-null int64  
persons     1728 non-null int64  
lug\_boot    1728 non-null int64  
safety      1728 non-null int64  
class       1728 non-null int64  
dtypes: int64(7)  
memory usage: 94.6 KB  
Everything is now converted to integer form.  
  
5) Select the predictor feature and the target variable

X = data.iloc[:,:-1]  
y = data.iloc[:,-1]

6) Train test split:

# split data randomly into 70% training and 30% test  
X\_train, X\_test, y\_train, y\_test = model\_selection.train\_test\_split(X, y, test\_size=0.3, random\_state=0)

7) Training/model fitting  
# train the decision tree  
dtree = tree.DecisionTreeClassifier(criterion='entropy', max\_depth=3, random\_state=0)  
dtree.fit(X\_train, y\_train)

8) Model parameters study:

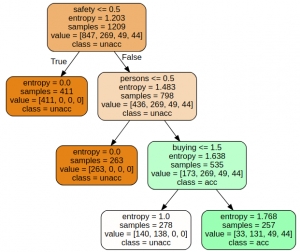
# use the model to make predictions with the test data

y\_pred = dtree.predict(X\_test)

# how did our model perform?

count\_misclassified = (y\_test != y\_pred).sum()

print('Misclassified samples: {}'.format(count\_misclassified))

accuracy = metrics.accuracy\_score(y\_test, y\_pred)  
print('Accuracy: {:.2f}'.format(accuracy))  
  
Misclassified samples: 96  
Accuracy: 0.82  
As you can see the algorithm was able to achieve a classification accuracy of 82% on the held out set. Only 96 samples were misclassified.  
  
   
  
Visualization of the decision graph:  
import graphviz  
feature\_names = X.columns  
  
dot\_data = tree.export\_graphviz(dtree, out\_file=None, filled=True, rounded=True,  
                                feature\_names=feature\_names,    
                                class\_names=class\_names)  
graph = graphviz.Source(dot\_data)    
graph